Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A system, comprising:

a plurality of enterprises having agents that process multi-agent cooperative business transactions, wherein each agent is configured to autonomously determine whether to complete a transaction;

a failure detector for detecting whether a failure is an inter-enterprise failure or an intra-enterprise failure;

an intra-enterprise failure handler coupled to the failure detector for performing failure recovery for intra-enterprise failures; and

an inter-enterprise failure handler coupled to the failure detector for performing failure recovery for inter-enterprise failures,

wherein the inter-enterprise failure handler includes

a scope determination module for identifying a failure recovery scope; and

a top-down logical undo module coupled to the scope determination module for undoing sub-transactions in an identified scope in a top-down manner.

2. (Previously presented) The system of claim 1 wherein the intra-enterprise failure handler includes

a scope determination module for identifying a failure recovery scope; and

a top-down logical undo module coupled to the scope determination module for undoing sub-transactions in an identified scope in a top-down manner.

3. (Original) The system of claim 2 wherein the scope determination module terminates at a closest ancestor of the failed transaction that is one of non-vital and associated with a contingency transaction.

4. (Previously presented) The system of claim 2 wherein in the scope determination module,

if a failed transaction is non-vital to its parent, continues with the parent transaction;

if a failed sub-transaction is determined to be a result of agent malfunction, the parent transaction delegates the sub-transaction to another agent; and

if a failed transaction is associated with a contingency transaction, re-tries the contingency transaction.

5. (Canceled).

6. (Previously presented) The system of claim 5 wherein in the top-down logical undo module,

if a transferred transaction has not started, does not perform failure handling on the transferred transaction;

if a transferred transaction is in progress, determines a rollback root of the transferred transaction; and

if a transferred transaction has completed, compensates for the transferred transaction.

7. (Previously presented) The system of claim 6 wherein in the top-down logical undo module,

if the rollback root of the transferred transaction is in progress, utilizes the rollback root as the root of recovery;

if the rollback root of the transferred transaction has committed to one of a parent and an ancestor, the parent is in-progress, and the rollback

root of the transferred transaction is one of non-vital and associated with a contingency transaction, utilizes a parent of the rollback root as the root of recovery; and

if the rollback root of the transferred transaction and a parent of the rollback root have committed, determines a highest committed ancestor of the rollback root of the transferred transaction and determines whether the rollback root of the highest committed ancestor is in progress;

if the highest committed ancestor is in progress, the highest committed ancestor of the rollback root of the transferred transaction is utilized as an extended rollback root;

if the highest committed ancestor is not in progress, the parent of the highest committed ancestor of the rollback root of the transferred transaction is utilized as an extended rollback root.

8.-16. (Canceled).

17. (Currently amended) A method, comprising:

starting a multi-agent cooperative business transaction between software agents of at least two enterprises;

detecting whether a failure is an inter-enterprise failure or an intra-enterprise failure;

when the failure is an intra-enterprise failure, performing failure recovery for the intra-enterprise failure; and

when the failure is an inter-enterprise failure, performing failure recovery for the inter-enterprise failure,

wherein performing failure recovery for the inter-enterprise failure includes identifying the failure recovery scope and undoing sub-transactions in the identified scope in a top-down manner.

18. (Original) The method of claim 17 wherein the step of when the failure is an intra-enterprise failure, performing failure recovery for the intra-enterprise failure includes

identifying the failure recovery scope; and

undoing sub-transactions in the identified scope in a top-down manner.

19. (Canceled).

20. (Original) The method of claim 19 wherein identifying the failure recovery scope includes the step of terminating at a closest ancestor of the failed transaction that is one of non-vital and associated with a contingency transaction.

21. (Currently amended) A system, comprising:

a plurality of computer-based enterprises having software agents that process multi-agent cooperative business transactions, wherein each agent is configured to autonomously make decisions that affect completion of a business transaction; and

a failure detector that detects if a business transaction fails; and

a failure handler coupled to the failure detector for performing failure recovery,

wherein the failure handler aborts a transaction if the transaction is determined to be non-vital to an associated parent transaction.

22. (Previously presented) The system of claim 21 wherein the plurality of software agents communicate based on a peer-to-peer mechanism that enables an agent to delegate a task to another agent and to copy information that is to be delegated to said other agent.

23. (Previously presented) The system of claim 21 wherein the plurality of software agents communicate based on a peer-to-peer mechanism that enables

each agent to log hierarchies of related transactions and to recover a logged transaction if a failure occur.

24. (Previously presented) The system of claim 21 wherein the plurality of software agents communicate based on a peer-to-peer mechanism that includes a preliminary commit stage and a final commit stage.

25. (Canceled).

26. (Previously presented) The system of claim 21 wherein, if a failure of a child transaction is determined to result from a malfunction of a first agent, the failure handler delegates a transaction from the first agent to a second agent.

27. (Previously presented) The system of claim 21 wherein the failure handler replaces a failed transaction with a contingency transaction.

28. (Previously presented) The system of claim 21 further comprising,

a first database that stores transaction data related to a first enterprise; and

a second database that stores transaction data related to a second enterprise;

wherein at least one agent is committed to the first database and at least one agent is committed to the second database.